BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

BACHELOR'S THESIS

Brno, 2021 Ondřej Nosek

BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS
ÚSTAV TELEKOMUNIKACÍ

CRYPTOGRAPHIC ESCAPE ROOM GAME
KRYPTOGRAFICKÁ ÚNIKOVÁ HRA

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR — **Ondřej Nosek**
AUTOR PRÁCE

SUPERVISOR — **M.Sc. Sara Ricci, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2021

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

Bachelor's Thesis

Bachelor's study program **Information Security**

Department of Telecommunications

Student: Ondřej Nosek ***ID:*** 203449

Year of study: 3 ***Academic year:*** 2020/21

TITLE OF THESIS:

Cryptographic Escape Room Game

INSTRUCTION:

At first the student will study state-of-the-art an interactive web application and escape room game.
Then the student will create an escape room game based on cyber security challenges (e.g. cryptography, network security, ...).

RECOMMENDED LITERATURE:

[1] SCHREUDERS, Z. Cliffe, et al. Security Scenario Generator (SecGen): A Framework for Generating Randomly Vulnerable Rich-scenario VMs for Learning Computer Security and Hosting CTF Events. In: 2017 USENIX Workshop on Advances in Security Education (ASE 17). 2017.

[2] Menezes AJ, Katz J, Van Oorschot PC, Vanstone SA. Handbook of applied cryptography. CRC press; 1996 Oct 16.

Date of project specification: 1.2.2021 ***Deadline for submission:*** 31.5.2021

Supervisor: M.Sc. Sara Ricci, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
Chair of study program board

ABSTRACT

The thesis deals with the implementation of the escape room game in the form of a web application. The topic of each room is cryptography in many forms. Concretely it is modular arithmetic, a system for data encryption, including its particular algorithms, as Advanced Encryption Standard is, or network security basics.
The main goals of the thesis are to get acquainted with the topic of escape games, web applications, and the realization of web applications. The escape game contains a total of four rooms. The thesis describes the choice of technologies on which the application will be built and the implementation of individual rooms, including the possibility of solving tasks. In the end, it summarizes the achieved results and goals.

KEYWORDS

Cryptography, Escape Room Game, Git, Gradle, Java, JavaServer Faces, Spring Framework, Web Application

ABSTRAKT

Tato bakalářská práce se zabývá implementací únikové hry ve formě webové aplikace. Tématem jednotlivých místností je kryptografie v mnoha podobách. Konkrétně se uživatel aplikace zabývá modulární aritmetikou, systémem pro šifrování dat včetně jeho menších částí, jako je u Advanced Encryption Standard, nebo také základy síťové bezpečnosti.
Hlavními cíli bakalářské práce je seznámení se s tématem únikových her, webových aplikací a realizace webové aplikace. Úniková hra obsahuje celkem čtyři pokoje. Práce popisuje volbu technologií, na kterých aplikace bude postavena, a postup implementace jednotlivých místností včetně možností řešení úkolů. V samotném závěru pak shrnuje dosažené výsledky a cíle.

KLÍČOVÁ SLOVA

Git, Gradle, Java, JavaServer Faces, kryptografie, Spring Framework, úniková hra, webová aplikace

NOSEK, Ondřej. *Cryptographic Escape Room Game*. Brno, 2021, 61 p. Bachelor's Thesis. Brno University of Technology, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Advised by M.Sc. Sara Ricci, Ph.D.

ROZŠÍŘENÝ ABSTRAKT

Zadáním bakalářské práce bylo nastudovat interaktivní webové aplikace a únikové hry. A na základě těchto informací vytvořit webovou aplikaci založenou na výzvách v oblasti kybernetické bezpečnosti, jako je např. kryptografie nebo síťová bezpečnost.

Práce je rozdělena do dvou hlavních částí, kterými jsou teoretické základy v kapitole Theoretical Background a technické zpracování aplikace v sekci Implementation, včetně postupu řešení jednotlivých úkolů.

V teoretickém úvodu se čtenář dozvídá základní informace o kryptografii a jakým způsobem ji v každodenním životě používá i bez vlastního vědomí. Jsou zde popsány dvě základní dělení kryptografických šifer – symetrické a asymetrické. Kapitola zmiňuje kryptografické služby, jimiž jsou šifrování zpráv, autentizace a integrita. Čtenář se seznamuje s termínem kryptologie, který se dělí právě na kryptografii a kryptoanalýzu.

Další teoretická část se věnuje webovým aplikacím – definici, vznik nebo využívaný protokol. Každá webová aplikace se skládá z několika jednotlivých webových stránek. Kapitola zmiňuje vývoj stránek od statických po dynamické a je doprovedena jednoduchými diagramy průběhu komunikace mezi klientem a serverem a zpracování dynamického obsahu na straně serveru. V kapitole je poznamenáno také aktuální populární přístup k webovým aplikacím typu single-page, který využívá službu např. mj. REST API na straně serveru, čímž aplikace přenechává zobrazování veškerého obsahu na klienta.

Java je název následující sekce zaměřující se na stejnojmennou platformu. Čtenář se dozví základní informace, jako je vznik, nezávislost na systémové platformě nebo popularita. Krátký odstavec se věnuje fungování Javy a jakým způsobem se stává multiplatformní. Ten je poté následován dělením Javy na dvě základní edice, tedy Standard a Enterprise Edition. Edice doplňuje Spring Framework.

Čtenáři doplní nebo osvěží znalosti verzovací systémy (VCS), kde se srovnává Subversion, neboli SVN, a Git. Z uvedených statistik vyplývá, že Git má na trhu majoritní podíl, jenž tvoří až 90 procent uživatelů, firem a projektů. Tabulka zde popisuje výhody a nevýhody obou VCS, což vysvětlí důvody popularity Gitu. Přiložená ukázka větvení pak ukazuje možnost používání Gitu.

Kapitola nástrojů automatizovaného sestavování (Build Automation Tool) zde porovnává rozdíly mezi Gradle a Maven, který je, jak z textu vyplývá, zaužívanější. Z přiložené srovnávací tabulky však vyplývá Gradle v mnoha ohledech jako lepší a vhodnější.

A teoretické základy uzavírá kapitolka o Lomboku. Popisuje zde, že se jedná o knihovnu pro Javu, která ulehčuje práci s objekty, jako je generování getterů a setterů, konstruktorů a mnoho dalšího. Přiložený výpis ukazuje přehlednost kódu s Lombokem a bez něj.

Ve druhé hlavní části, implementaci, se v první řadě shrnuje tzv. Tech Stack, v českém jazyce jako projektové technologie. Kromě využitých technologií pro projekt se zde zmiňuje i vývojové prostředím, kterým je IntelliJ IDEA, s tzv. pluginami. Čtenář se zde také dozví konečný výčet knihoven používaný v projektu.

Jak se v úvodu realizace popisuje, před vývojem přichází analýza projektu. Ústředním tématem je zde především rozdělení aplikace do několika vrstev, konkrétně do čtyř, a to prezentační, servisní, persistentní a datové. Doprovodný obrázek ukazuje komunikaci mezi jednotlivými vrstvami. Upozorňuje se zde na fakt, že vyšší vrstva může komunikovat pouze s nižší vrstvou a vrstvy se nesmí přeskakovat.

V prezentační vrstvě je zmíněn návrhový vzor Model-View-Controller, který se používá právě pro uživatelské rozhraní. Kapitola poté informuje, že kontrolér se může skládat z několika komponent, které mají vlastní odpovědnost. K tomu jsou dodány informace, jakým způsobem jsou injectovány komponenty do sebe, co je to scope (česky rozsah) a jaký má vliv na fungování komponenty.

Kapitola o persistentní vrstvě ukazuje fungování Java Persistence API a vysvětluje fungování nezávislosti na databázi.

Následující odstavce se věnují vytvoření projektu pomocí webové aplikace Spring Initializr a následuje ukázka loga s lehkým vysvětlením, za pomoci jakých nástrojů a technologií logo vzniklo.

Nápis Room Alpha značí jediné, a to popis v pořadí první místnosti webové aplikace. Čtenáře uvede do problému a ukáže design místnosti, který je laděn do prostředí školní třídy. Jednoduchými dedukcemi vyvodí, že pro řešení úkolů bude zapotřebí modulární aritmetika. Avšak při popisu, jak postoupit do další místnosti, hned ukazuje, že existuje i další řešení, kterým je útok hrubou silou.

Kapitola pokračuje řešením prvního úkolu s hodinami a hledáním výsledku za pomoci kongruentní rovnice. Řešení je podpořeno několika vysvětlujícími rovnicemi. Po zjištění výsledku se kapitola zakončuje podmínkami generování čísel, aby bylo úkol možné vyřešit.

Následuje matematické řešení Eulerovy funkce, kterým byl druhý úkol v první místnosti. Je zde popsán celý postup výpočtu s konečným výsledkem. Stejně jako v předešlém případě, i zde je nutné vytvořit hranice pro generování čísla, které jsou odůvodněny na přiloženém grafu. Generování čísla ještě doprovází jednoduchý algoritmický diagram. Zadáním správného výsledku do kódové čtečky se uživatel dostává do další místnosti a čtenář do další části práce.

Kapitola Room Bravo popisuje design místnosti, která vypadá jako pracovní pokoj se značně zájmu přitažlivým obrazcem. Několika myšlenkovými kroky dochází k závěru, že na zdi se nachází matice 4×4 a jedná se o parciální procesy šifrovacího algoritmu AES. Po inverzním výpočtu se dochází k výsledku. Následují odstavce o generování slova a šifrování, neboť stojí za zmínku použití návrhového vzoru

Továrna, neboli Factory. K návrhovému vzoru je dodán jak diagram, tak i výpis implementace v jazyku Java.

V další kapitole čtenáře na obrázku přivítá bezpečnostní hlídač, jenž se nachází ve třetí místnosti této únikové hry. Následující text opět seznamuje čtenáře s prostředím místnosti, kterým je chodba v galerii. Poté popisuje postup řešení úkolu se zakódovaným obrázkem v Base64 a následně dešifrování certifikátu. Zmiňuje se zde také o sociálním inženýrství a možném obejití úkolu. Prokládané ukázky algoritmů vysvětlují implementační problematiku.

Kapitola o místnosti s názvem Delta čtenáře informuje o tom, že se jedná o poslední minihru. Rovněž popisuje prostředí místnosti, kde za zmínku stojí terč s jedinou netrefenou šipkou do něj. Nakonec informuje, že se jedná o úkol s nastavením firewallu. Před řešením se zde objevuje návrhový vzor fasáda. Na jednoduchém příkladu pak matematicky vysvětluje fungování zástupné masky (wildcard mask). V několika dalších krocích vysvětluje postup řešení. Místnost uzavírá několika odstavci o kontrolních testech, které ověřují správné nastavení firewallu.

Na závěr implementační části se práce zmiňuje o nasazení SSL certifikátu pro bezpečné spojení mezi klientem a webovým serverem.

V teoretické části práce byly nastudovány možnosti webových aplikací. Následně byla naimplementována kryptografická úniková hra ve formě webové aplikace, jak bylo požadováno dle zadání. Webová aplikace umožňuje lehký přístup uživatelů z jakéhokoli zařízení s využitím standardních webových prohlížečů, jako jsou Safari, Mozilla Firefox, Google Chrome nebo Opera. Jednotlivé úkoly, které se nacházejí v místnostech, se mohou jevit jako složité. Na druhou stranu, některé místnosti je možné vyřešit i jiným způsobem než standardní cestou. Uživatelé si také mohou jednotlivé úkoly stanovit jako osobní výzvu jako například v případě puzzle boxů.

DECLARATION

I declare that I have written the Bachelor's Thesis titled "Cryptographic Escape Room Game" independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author I furthermore declare that, with respect to the creation of this Bachelor's Thesis, I have not infringed any copyright or violated anyone's personal and/or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno ....

author's signature

ACKNOWLEDGEMENT

In the first place, I would like to thank my supervisor M.Sc. Sara Ricci, Ph.D., for leading the thesis, help, support, and professional suggestions and comments. I also thank my girlfriend and friends for their infinite support and patience. I really appreciate it.

Contents

List of Figures

Listings

Introduction

Cryptography is a standard part of our lives without even noticing it. Trying to protect secrets began thousands of years ago. We could say that modern cryptography [4] began during World War II and continues with a vast usage of the Internet.

Many protocols [1] [2] designed in 70s, 80s do not include any security aspects, for example, `HTTP`, `SNMP`, `FTP`. Later in the 90s, many standards were redesigned to implement security components. And often not so successfully. In 1993 [1] `SNMP v2c` standard was released with improved security, which was community string in plain text. Eleven years later, they released version 3 with authentication and privacy encryption elements that meet the nowadays requirements and standards.

The thesis focuses on parts of cryptography, which are modular arithmetic, cryptography protocols, or network security. The examples are presented in the form of a web exit room game application. The user as a player has to fulfill the tasks to finish the game in 4 rooms.

The application is developed on the Java platform with the Spring framework. Application HTML GUI is written in JavaServer Faces technology with PrimeFaces framework.

The Theoretical Background chapter describes basic information about cryptography, web applications, escape room games, programming language, and accompanying tools and libraries for the software development, which were used during the implementation of the application.

The Implementation chapter focuses on the realization of the application with part of the analysis, application layers, and worth mentioning algorithms such as design patterns. The chapter then continues with thoroughly described rooms and their challenges with ways of solutions.

At the end of the bachelor's thesis, Conclusion evaluates the achieved goals and results of the thesis. After the summary chapter, there are appendices and other additions to the thesis.

1 Theoretical Background

1.1 Cryptography

Cryptography is all around us. People do not even realize using it. According to Google Transparency Report [3], from 14th November 2020, 98 percent of pages that people visit using Google Chrome use HTTPS protocol. In every single HTTPS communication, cryptography is used.

Cryptography [4] is studying methods and resources to provide cryptographic services. Some of the basic services are described below with everyday usage. The services are based on mathematical problems or time-consuming finding a solution.

The most popular provided service is message encryption (secrecy), which is often used to encrypt messages between two subjects (people, companies, or devices). The messages should not be readable by third parties involved in the communication.

Another important cybersecurity feature is authentication. In fact, authentication as a service provides certainty of communication with the right subject. This service is also used daily. Disruption of authenticity while using HTTPS shows a warning that the connection to the server is not private, as shown in Figure 1.1.

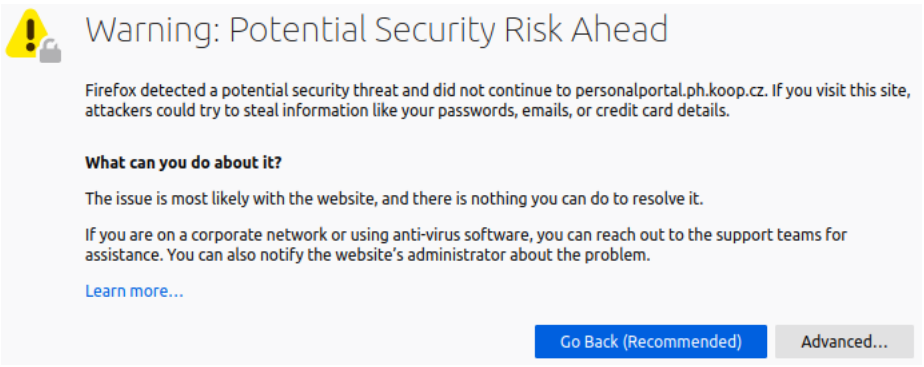


Fig. 1.1: Disruption of authenticity

Moreover, the integration service allows electronic documents to record the state with something like a fingerprint. When the document changes, the fingerprint changes too. Figures Figure 1.2 and Figure 1.3 show the difference between fingerprints of the original and the changed text. It is evident at first glance that the fingerprints are entirely different.

```
98334b67cd47213977ee1943bade5926...
```

Fig. 1.2: Example of fingerprint for the original text:
Please send the money here: 100245/2300

```
e948e33fa19e53f6cb802b17a90f686b...
```

Fig. 1.3: Example of fingerprint for the changed text:
Please send the money here: 204812/1100

Cryptography consists of two methods of encryption – symmetric cryptography and asymmetric cryptography. Symmetric cryptography can be split into two types, which are Block Cipher and Stream Cipher. Block cipher [4] is processed block by block, and stream cipher is processed byte by byte or bit by bit. Comparision is shown in the table Table 1.1

The difference between symmetric and asymmetric cryptography is that symmetric cryptography uses one key to encrypt data and the same one to decrypt encrypted data. Algorithms using symmetric cryptography are, for example, AES, DES, 3DES, RC4. Algorithms using asymmetric cryptography are, for example, RSA, Diffie-Hellman, El Gamal. Figure 1.4 describes asymmetric cryptography.

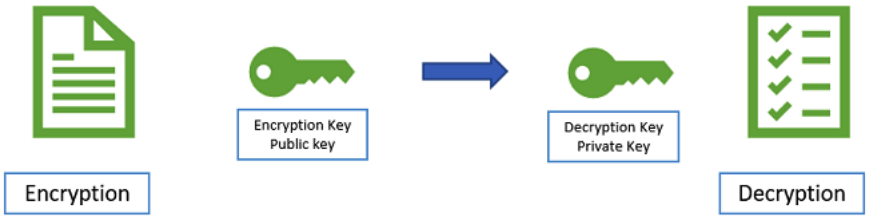


Fig. 1.4: Asymmetric Encryption [5]

As described above, generally, cryptography is studying how to save data. There also exists a discipline that is studying vulnerabilities of cryptographic systems, i.e., cryptoanalysis. According to the knowledge of the discipline, cryptographic systems can improve their functionality. However, cryptoanalysis can also be used illegally to get secured data. This attack is known as hacking. Many insecure cryptographic algorithms should not be already used. Examples are WEP [6] and MD5 [7] in specific scenario.

Another discipline named Steganography uses media such as pictures, videos, and music to hide data and information. All these three disciplines are part of the scientific field – Cryptology [4].

Tab. 1.1: Comparison of stream cipher and block cipher, [8]

	Stream Cipher	Block Cipher
Processing	Byte	Block of bytes in size of 64 bits or more
Decryption	Easier	Harder
Speed	Faster	Slower
Algorithm	RC4	AES, DES

1.2 Web Application Games

A web application [9] is software running on a web server that consists of a collection of servlets, HTML pages, classes, and other resources. A web application is rooted at a specific path within a web server.

Web applications often use the HTTP or more modern secured HTTPS protocol. The web pages which the application contains can be static or dynamic. Static pages are usually in HTML, while dynamic pages consist of adapted content for the specific user.

The beginning of using dynamic content started the Web 2.0 era. The dynamic pages can be built on many platforms, such as PHP, Java, C#, or Python. A favorite variant in recent years is using an HTML/JS template and an API, for example, REST API. This type of web application is well-known as a single-page application.

Figure 1.5 shows that the Web Server only resends the HTML page without any processing, while in Figure 1.6, the Web Server has its Interpreter, which processes the content before sending it to the Client.

This project uses dynamic pages, which provide dynamically generated values for the tasks.

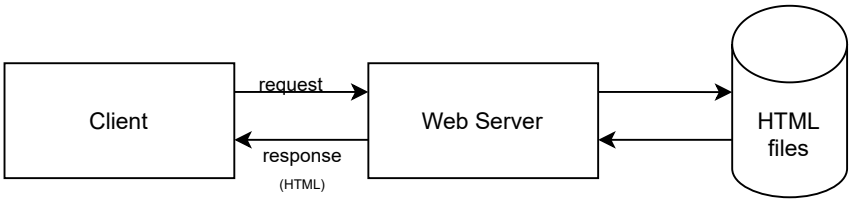


Fig. 1.5: Static web page

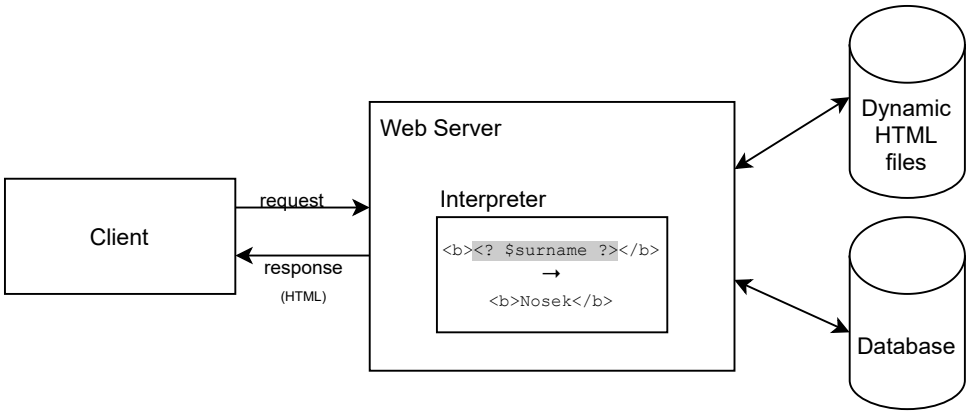


Fig. 1.6: Dynamic web page

1.3 Escape Room Games

The first escape game [10] began in Japan in 2007, then reached the United States, where it experienced a great boom by 2018. Escape games are often played in teams and are popular with companies as a team-building activity. The tasks stand on the border between puzzle and theater.

The game's course [10] consists of closing the group to a room in which they have to solve tasks and puzzles. Some challenges may require multiple player cooperation to solve tasks, which is used in team building.

According to the experiment [10], escape games can help with education. In an escape game on a chemical topic, 70 percent of participants said it helped them better understand the issue.

1.4 Java

Java [11] is a high-level programming language with the first stable release in 1996 by James Gosling. It is a multi-platform object-oriented language. According to Business Insider, Java appeared in 2019 as the second most popular programming language in the world (based on statistics of GitHub) [12]. The newest stable version [13] is Java SE 16. Java S6 15 was released on 16 March 2021.

Bytecode and JVM

To run the Java code, it is necessary to compile it in Java bytecode. Java Virtual Machine can run this code. JVM is developed for almost every mainstream platform, which generally makes Java a multi-platform. Users often use Java Runtime Environment (also known as JRE).

Some platforms provide direct hardware support for running Java bytecode, which is often microcontrollers.

Editions

Java Standard Edition

Java Standard Edition provides the core functionality of the Java programming language. Java SE contains from basic types and classes to high-level classes aimed at solving specific problems.

Java SE is used for basic applications that run on client's devices, so-called client-side applications. The applications can be games, small utilities also more complex programs.

Jakarta Enterprise Edition

Jakarta Enterprise Editions is part of the Java Platform. Before named as Java 2 Enterprise Edition (J2EE), later as Java Enterprise Edition (Java EE), and now as Jakarta Enterprise Edition (Jakarta EE). Generally shortly as JEE.

JEE is built on Java SE with extending specifications for enterprise applications. The specifications contain a defined API. The enterprise applications may consist of many parts that JEE provides.

Spring Framework

The Spring is [14] a lightweight solution for enterprise applications. Spring is modular and allows users and developers to use only necessary parts. According to Spring documentation [14], they do not get into the role of competitors to JEE. Spring is complementary to Java EE. However, Spring can run enterprise applications without JEE.

1.5 Version Control System

There are many systems to handle the versioning of developed applications. Versioning does not apply only to software development but can be used on every data saved on any device. This section focuses on software version control.

Version Control System (VCS) [15] controls who, when and what was changed in the software. According to this feature, VCS can reverse each file to its creation. Reversion can be applied when an incorrect behavior of the software appears with a developer's change.

VCS is also very popular, thanks to the ability to collaborate with many developers. Each developer works on his/her local instance (clone) of the repository. The repository is a system of folders and files of the software. When the change is done, changes are uploaded from the local repository to the remote repository. While uploading, conflicts can occur. Conflict means that two or more developers made changes in one or more files during developing developer's changes.

Figure 1.7 depicts creating two new branches, representing two independent developers, from the main branch. Both developers made changes to the Foo.java file. The first branch merges to the main branch effortlessly, while the second one must resolve conflicts because the first developer's changes have not been reflected in the second developer's branch. The second option, marked in blue, indicates the continuous connection of the main branch and the resolution of conflicts before the merge with the main branch.

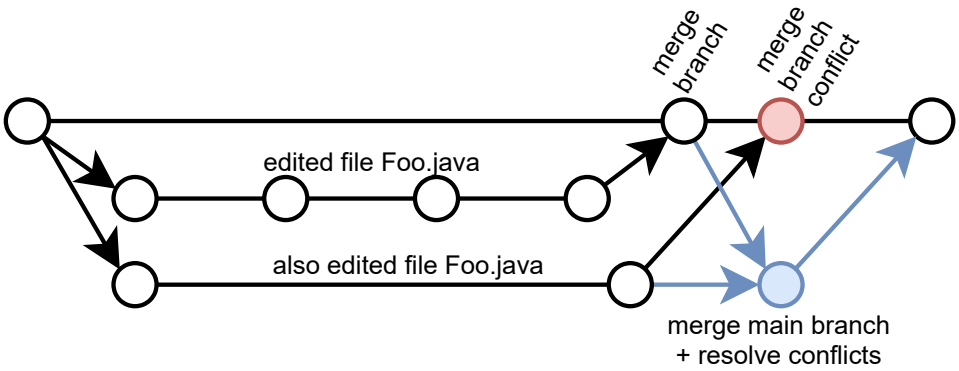


Fig. 1.7: VCS conflict

Git

Git is a free and open-source version control system and is the most used VCS. According to Open Hub statistics [16], there is 72 percent of project repositories based on Git. Another 23 percent is Subversion (SVN), and 5 percent is another. And according to the Stack Overflow Developer Survey [17], almost 90 percent of developers use Git.

Git development [15] began in 2005 after arising issues with BitKeeper (VCS used at the time) between Linux kernel developers. BitKeeper ceased to be free to use the software. Linus Torvalds, the creator of Linux, needed VCS, but none of the available systems did not meet his requirements. So Torvalds started developing his own.

Based on the arguments described in Table 1.2, it was decided to use Git as VCS.

Tab. 1.2: Git vs. Subversion

Git	Subversion
Distributed	Centralized
Commits are saved locally	Commits are sent to remote repository
Easy branching	Hard branching
90% professionals' support	5% professionals' support

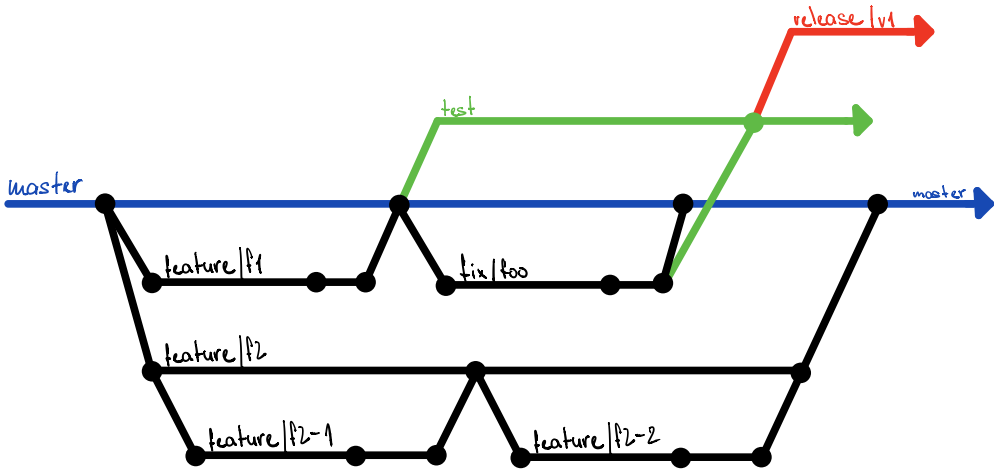


Fig. 1.8: Example of using Git branches

According to comparison of Git and SVN it was decided to use Git.

1.6 Software Testing

Software testing provides product quality information. The output of testing is a report with discovered software bugs, unpredictable behavior, security risks, completion of requirements, and other information. Testing begins with settings goals and scope of the tests (what to test – which parts).

Software testing is performed in many ways – manual testing, semi-automatic testing, and automatic testing. Manual tests are very good for one-time tests or short-term applications, environment exploration (for black-box testing), or irregular procedures that would be difficult to automate. But many tests can be automated. A combination of automated and manual testing are semi-automatic tests – for example, web page testing: user registration and login can be automated, and the rest of the procedure can be done manually.

1.7 Build Automation Tool

Build automation [19] is an automated software building tool that includes compiling source code into the byte code (in the case of Java), packaging compiled code, and running automated tests. Some tools also [20] provide dependency management.

For Java projects, there are several tools that provide build automation. According to a survey published in October 2018 by Snyk, the two most popular tools are `Maven` with 60 % of developers support and `Gradle` with 19 % support [21].

The application uses Gradle as a building tool. Comparison is described in Table 1.3.

Tab. 1.3: Maven vs. Gradle [22] [23]

	Maven	Gradle
Performance	Worse – build is mandatory	Better – uses cache, Incremental build and Compiler Daemon; Avoids to build
Dependency Management	Overrides only by version	Customizable dependency selection and substitution rules
Separation	None	Custom structure – e.g. unit and integration tests
Customization	Not so customizable	Very customizable
Android Applications	–	Supported by Google

1.8 Lombok

Lombok is a Java library for generating many methods like getters, setters, hashCode, equals using annotations (`@Getter`, `@Setter`, `@Data` etc.) and many others.

```
// without Lombok
public class UserData {

    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        SomeData someData = (SomeData) o;
        return name.equals(someData.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name);
    }
}

// with Lombok
@Data
public class UserData {
    private String name;
}
```

2 Implementation

The thesis focuses on the implementation of a point-and-click game. Every project consists of analysis, goal setting, and the implementation itself. Users can use the application for educational purposes or as their challenge, like puzzle boxes.

The following chapters describe individual parts of development and the final implementation of the application in the form of a description of each room. Additionally, the description of the rooms accompanies remarkably interesting facts and processes from the realization of the application.

The application stands on the Java 11 platform with the Spring Framework. Additional libraries are JavaServer Faces with PrimeFaces framework for web GUI, JoinFaces for integration with Spring Framework, log4j2 for logging, and Lombok for simplified work with classes.

IntelliJ IDEA [24] provides a complex, easy-to-use, and intelligent coding assistance with ergonomic design Integrated Development Environment (IDE). This IDE developed by Czech company JetBrains s. r. o., integrates not only Gradle by default and is open for additional plugins. An additional plugin for the development environment is Lombok, which integrates the IDE with the annotation processor described above.

2.1 Project Analysis

The application has been separated into four layers – `Presentation Layer`, `Service Layer`, `Persistence Layer` and `Data Layer`. Each layer has its own responsibility. The higher layer can use just and only the lower layer – Presentation Layer does not call any Persistence Layer API but uses just Service Layer. Figure 2.1 depicts the dependencies among the layers, which are described below in more detail.

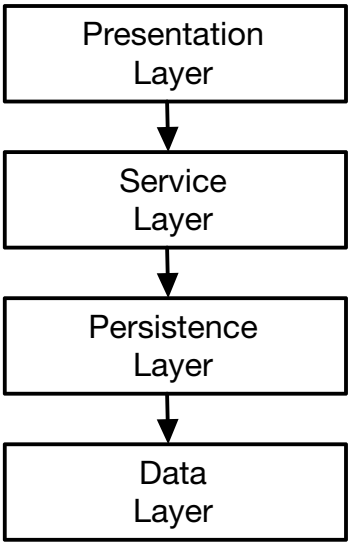


Fig. 2.1: Application layer architecture

Presentation Layer

The Presentation Layer uses Jakarta Server Faces API. It is the design of the application, and it is what the user sees. The layer's responsibility is to react to the user's actions like clicking on buttons, filling input boxes. Received data are collected, formulated, and sent to Service Layer, where the data are processed and retrieved to the Presentation Layer.

This layer is in the project in a separated package named `view`.

JSF is a Model-View-Controller framework. The MVC is a pattern used for user interfaces. The View is represented as `xhtml` files that contain HTML and JSF UI components, the model is a Java data object bonded with View, and the Controller is responsible for actions made by the user. The diagram of the MVC pattern depicts Figure 2.2.

The controllers may divide into several components which have their responsibility. These components are annotated by Spring `@Controller` and `@Scope`. The

first annotation causes that the Spring Framework will include in the CDI. Whenever any other Spring component requires the class, it can inject it using annotation `@Autowired`. The scope annotation handles the life of the component. Some of them are:

- `Singleton`: Each injection gets the same instance
- `Prototype`: Each injection creates a new instance
- `ApplicationScope`: Same as Singleton but for web application
- `SessionScope`: New HTTP session creates a new instance
- `RequestScope`: Each HTTP request creates a new instance

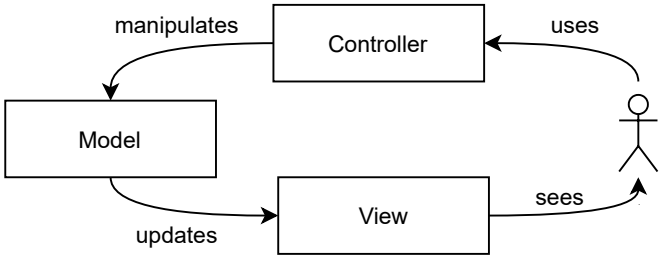


Fig. 2.2: Model-View-Controller pattern

Service Layer

The Service Layer takes care of all business processes in the application. It is used by the Presentation Layer and uses the Persistence Layer. Spring Framework components form the Service Layer.

In the project, the Service Layer is in a separated package named `service`. The only business process that the service layer handles is providing game statistics and scores. The services are annotated as `@Service` and injected using CDI with `@Autowired`, as well as in the Presentation Layer.

Persistence Layer

Persistence Layer uses Jakarta Persistence API, concrete implementation Hibernate. JPA consists of three parts which are: API, object-oriented mapping, and Jakarta Persistence Query Language.

Using this component makes the application's back-end independent of database type. Every database has its SQL differences. The SQL is used to select, insert, or generally manipulate the data in the database.

JPA uses its JPQL, which is translated into SQL based on database type. This causes that the back-end layer is database-type independent. Figure 2.3 shows JPQL

select, which is translated into SQL in Figure 2.4. The only table that the database contains is stats, which saves data with nicknames and time spent in the game.

```
SELECT u FROM User u WHERE u.id = 1
```

Fig. 2.3: JPQL selection query

```
SELECT * FROM users WHERE id = 1
```

Fig. 2.4: Translated SQL selection query

Data Layer

The primary purpose of the Data Layer is to store the application data and provide an API to manipulate the data. Data Layer is presented by MySQL database.

Game Progress

The application consists of four rooms. The progress in the game shows the diagram in Figure 2.5.

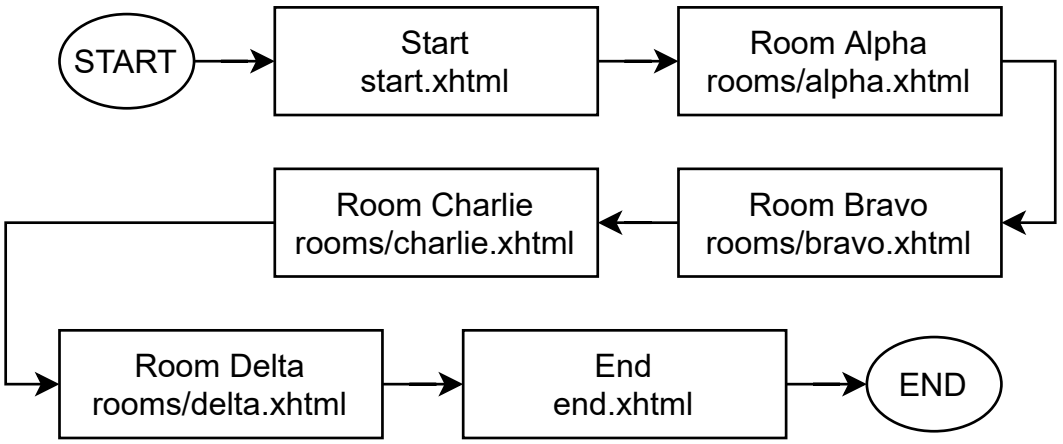


Fig. 2.5: Game progress

2.2 Creating a New Project

The Spring Framework has its project creator called Spring Initializr[1] shown in Figure 2.6. This tool allows users to generate projects with many Spring packages quickly. The tool enables users to choose the Spring version, Java version, build tool, or basic project settings such as name, artifact.

As described in the Implementation, the application project is Gradle with Java version 11. Spring is modular, so many modules can be selected for the project in the dependencies section, shown in Figure 2.7. For this application, Lombok was added. Any other dependencies can be added later.

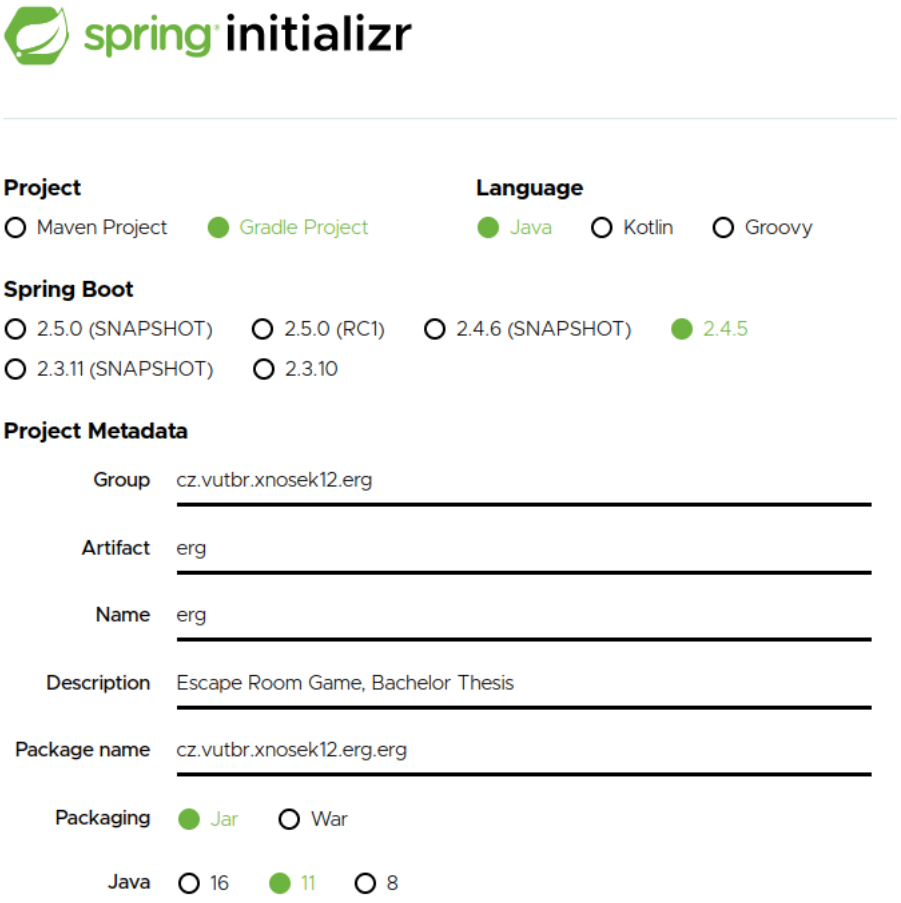


Fig. 2.6: Spring Initializr

[1]Spring Initializr - https://start.spring.io/

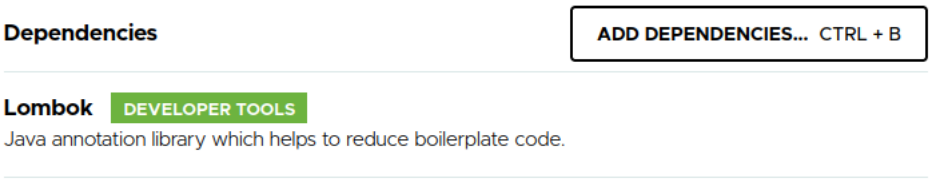


Fig. 2.7: Spring Initializr dependencies

All project settings are located in `build.gradle`, includes dependency management, Java compiler version, and more sophisticated actions and commands. Other dependencies were added:

- JoinFaces, version 4.0.1
- PrimeFaces, version 6.2
- Apache Commons Lang 3, version 3.0
- Spring Data, version, version 2.4.5
- MySQL Java Connector

2.3 Logo

The logo presented in Figure 2.8 and any other design were created using iPad Air 2020 with Apple Pencil 2 using the application Notability. The logo contains three primary characters – a lock as security, a chip, and a terminal command with the purpose of the application.

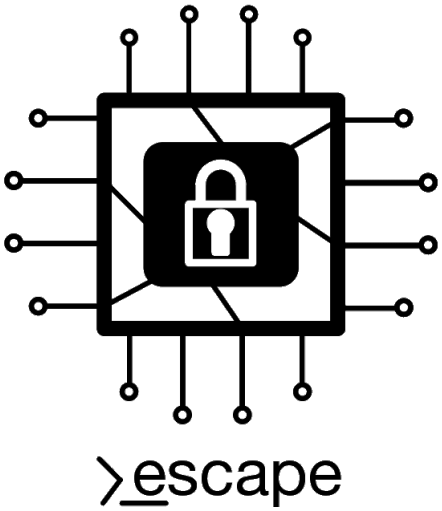


Fig. 2.8: Logo

2.4 Rooms

2.4.1 Room Alpha

As the name says, room Alpha is the first room of the application, and it is designed as a school classroom. At first glance at Figure 2.9, the room includes an unusual clock, a door with a PIN code keyboard, and a phi function on the board.

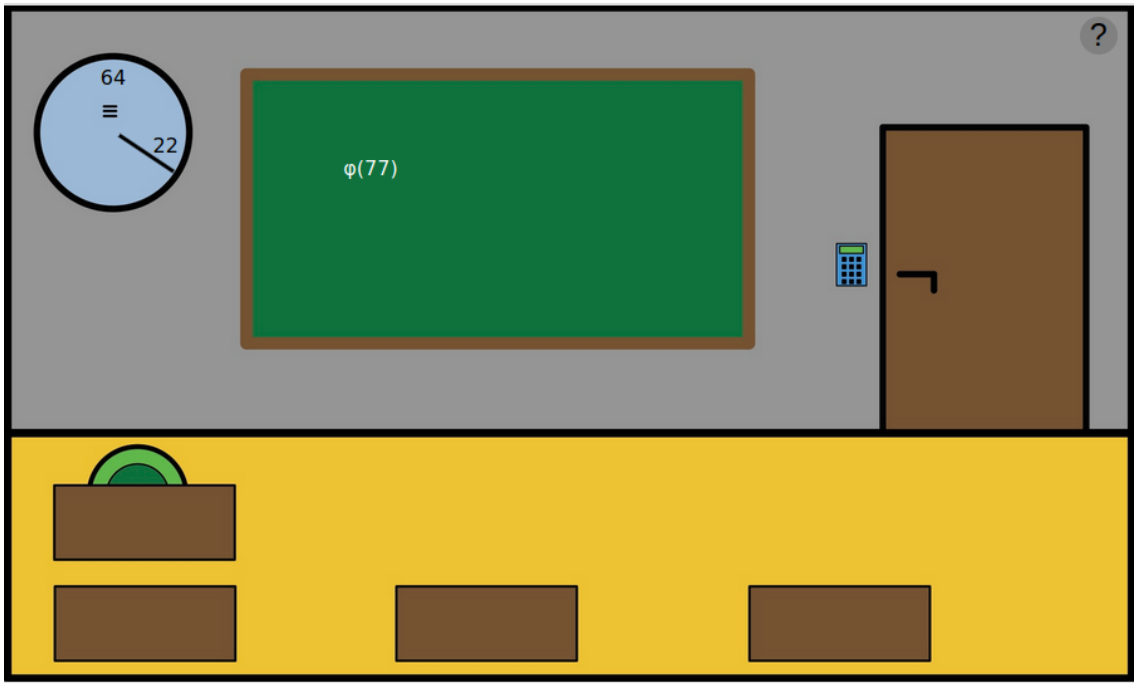


Fig. 2.9: Room alpha

Many hints have already suggested that the room is about modular arithmetic. There are two quests to achieve the code as described below.

PIN Code Keyboard

For entering the door and proceed to the next room, it is necessary first to unlock the door with a PIN. The user can find out the PIN code by solving tasks or, as in cryptography, by brute force attack, which is a possible but a challenging way.

However, the code is four digits long, which can be found when trying to write a random PIN. So the range is $0000 \rightarrow 9999$, which means there are 10^4 combinations.

The PIN code is split into two parts. The first one is calculated from the clock, and the second one is calculated from the phi function. The dialog for entering the PIN code is described in Figure 2.10.

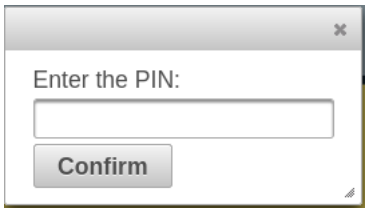


Fig. 2.10: PIN code verification

The Clock

A brute force attack is not always the best way. It takes a lot of time and resources. In this case, it is better to guess the PIN code in the standard way. That is, firstly, solve the clock.

No one misses the fact that the clock is not an ordinary clock for displaying time. The watch also contains a familiar symbol of congruence (≡) for cryptographers. The numbers on the clock are randomly generated.

To solve the watch-game, the solver has to make a congruent equation. After a short Google search with properly chosen keywords (*modular arithmetic*), a trendy page[2] describes basics about congruence with an illustrated demonstration with a clock. The equation is:

$$x \equiv b \pmod{n}, \tag{2.1}$$
$$x = kn + b, \tag{2.2}$$
$$k \in \mathbb{Z}, \tag{2.3}$$

where n is modulus, which is, in a case of clock, $n = 12$, analogous to the described case, it is $n = 64$. For b the number is on a clock the minute hand, which, for the demonstration, is $b = 22$. So the congruence equation is:

$$x \equiv 22 \pmod{64}, \tag{2.4}$$
$$x = 64k + 22. \tag{2.5}$$

For the range $0 \to 99$, the $x \in \{22, 86\}$.

[2]Wikipedia Modular arithmetic – `https://en.wikipedia.org/wiki/Modular_arithmetic`

Because the puzzle result can have only two digits, it is necessary to generate the numbers correctly. The second condition was not to generate too many numbers. Moreover, the last one was to create the number other than the remainder (b). Therefore the following rules have been estimated:

$$b < n, \tag{2.6}$$

$$b + n < 100, \tag{2.7}$$

$$n > 33, \tag{2.8}$$

$$x > b \tag{2.9}$$

where x is the generated number, n is the modulus and b is the remainder from the congruence equation above in Equation 2.1. According to the rules, the maximal size of the result set can be two: $x \equiv 1 \pmod{34}$, then $x \in \{34, 69\}$.

According to the rules, the result of the described case is $x = 86$.

Euler's Phi Function

The second part of the PIN code is from the calculated example on the board. The example is, as the title suggests, the Euler's Totient function:

$$\varphi(n) = n \prod_{n|p} \left(1 - \frac{1}{p}\right), \tag{2.10}$$

$$\varphi(n) = p_1^{k_1 - 1}(p_1 - 1) \cdot p_2^{k_2 - 1}(p_2 - 1) \cdot \ldots \cdot p_q^{k_q - 1}(p_q - 1), \tag{2.11}$$

$$\varphi(n) = n \cdot \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_1}\right) \cdot \ldots \cdot \left(1 - \frac{1}{p_q}\right) \tag{2.12}$$

where p_i is distinct prime, which divides n, k_i is number (quantity) of p_i, and q is number of distinct primes. Therefore

$$n = \prod_{i=1}^{q} p_i^{k_i}, \tag{2.13}$$

$$n = p_1^{k_1} \cdot p_2^{k_2} \cdot \ldots \cdot p_q^{k_q}. \tag{2.14}$$

For the described example $\phi(77)$:

$$77 = 7^1 \cdot 11^1, \tag{2.15}$$

$$\varphi(77) = 7^{1-1}(7 - 1) \cdot 11^{1-1}(11 - 1), \text{ or} \tag{2.16}$$

$$\varphi(77) = 77 \cdot \left(1 - \frac{1}{7}\right) \cdot \left(1 - \frac{1}{11}\right), \tag{2.17}$$

$$\varphi(77) = 60. \tag{2.18}$$

A faster and easier way to get the number is to try to search for an online calculator. With a few targeted keywords, one of many pages can be found[3].

As well as the first number from the PIN, the Phi function must have no more than two digits, which means making a limitation. Firstly, it is necessary to find out when the function exceeds 99.

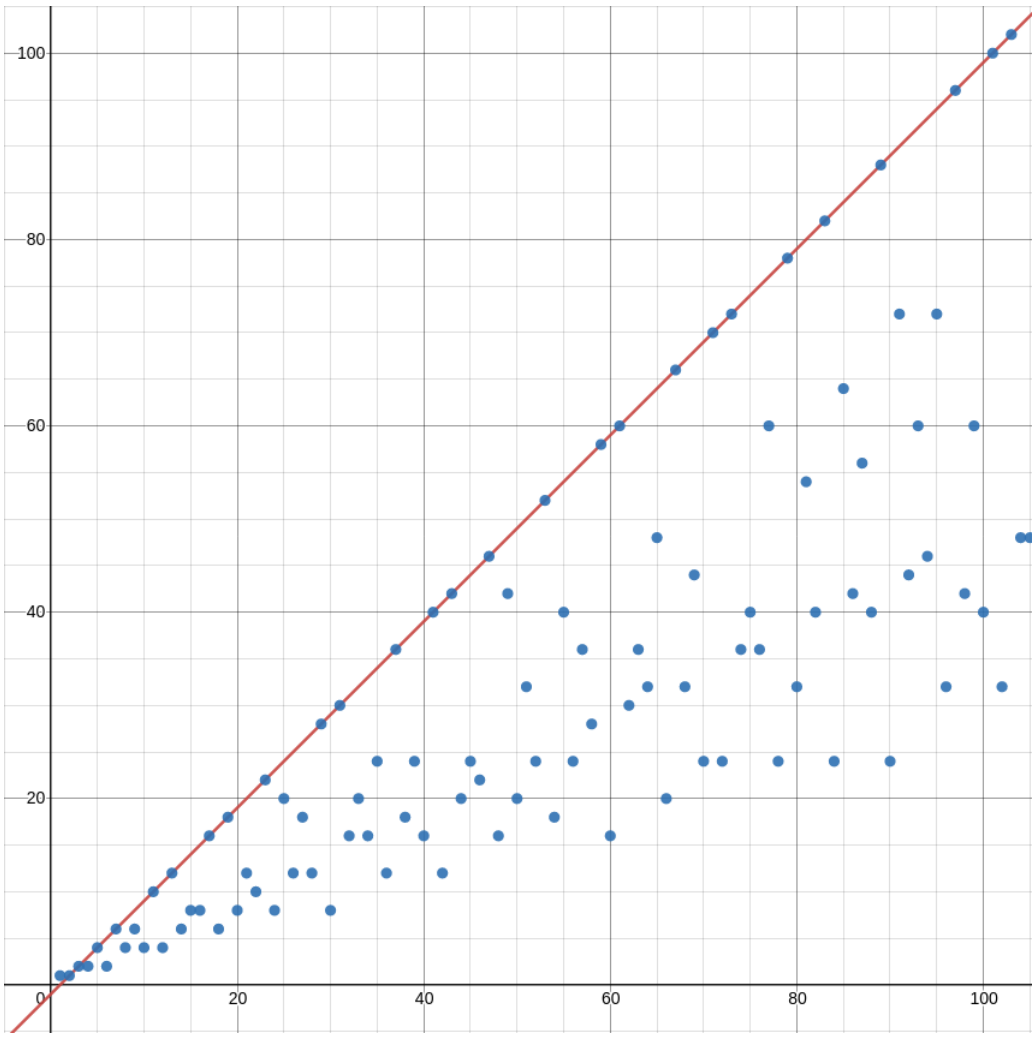


Fig. 2.11: Euler's Phi function graph

In the Figure 2.11, the graph shows values of Phi function described as $\varphi(x) = y$. The blue dots are the function results. The red line is the function $f(x) = n - 1$ and describes the maximal result of the Phi function.

As can be seen from the graph, the first number that exceeds 99 is number 101. So the limitation is: $x < 101$. An additional check is implemented in the code,

[3]Euler's Totient Calculator - Online Phi Function – `https://www.dcode.fr/euler-totient`

which performs the function of careless handling of the code. Figure 2.12 describes the algorithm.

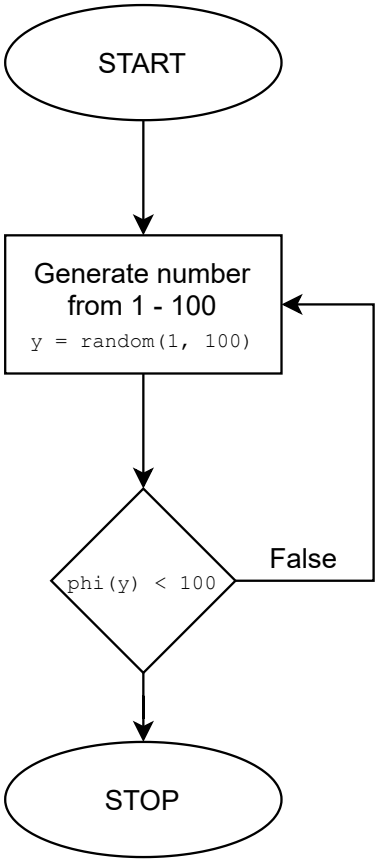


Fig. 2.12: Generating algorithm

Both numbers have been calculated; they are 86 and 60. Together they give `8660`. After entering the password into the code lock, the application redirects the user to the second room.

2.4.2 Room Bravo

Room Bravo is the second room of the application. It is located in someone's bedroom or office. The most striking figure is on the wall, which is, in fact, a 4×4 matrix. However, the only active element of this room is a door handle, which requires a word after clicking on it.

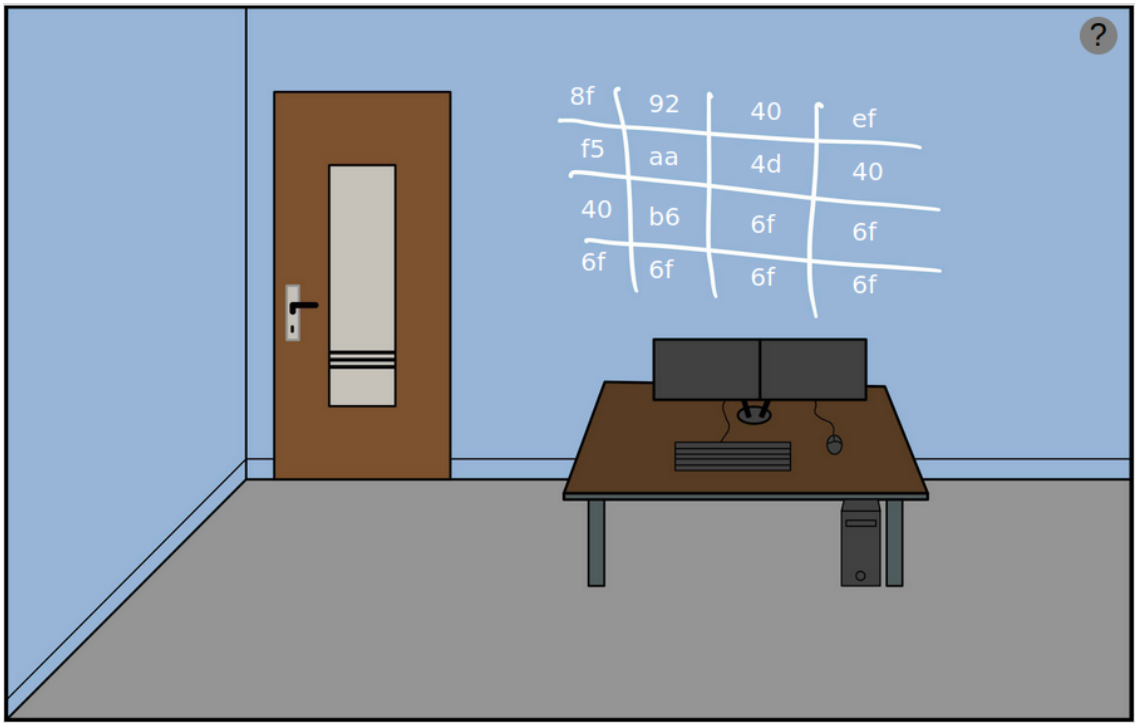


Fig. 2.13: Room bravo

There are a few popular cryptography algorithms that use a 4×4 matrix. One of the few is AES. In this algorithm, there are four encryption methods: *ShiftRows*, *MixColumns*, *SubBytes*, and *AddRoundKey*.

$$\begin{pmatrix} 8f & 92 & 40 & ef \\ f5 & aa & 4d & 40 \\ 40 & b6 & 6f & 6f \\ 6f & 6f & 6f & 6f \end{pmatrix} \rightarrow \begin{pmatrix} 8f & 92 & 40 & ef \\ 40 & f5 & aa & 4d \\ 6f & 6f & 40 & b6 \\ 6f & 6f & 6f & 6f \end{pmatrix} \sim \begin{pmatrix} ? & ? & @ & \ddot{\imath} \\ @ & \tilde{o} & ? & M \\ ? & ? & @ & \H{u} \\ ? & ? & ? & ? \end{pmatrix}, \tag{2.19}$$

$$\begin{pmatrix} 8f & 92 & 40 & ef \\ f5 & aa & 4d & 40 \\ 40 & b6 & 6f & 6f \\ 6f & 6f & 6f & 6f \end{pmatrix} \rightarrow \begin{pmatrix} 73 & 74 & 72 & 61 \\ 77 & 62 & 65 & 72 \\ 72 & 79 & 06 & 06 \\ 06 & 06 & 06 & 06 \end{pmatrix} \sim \begin{pmatrix} s & t & r & a \\ w & b & e & r \\ r & y & ? & ? \\ ? & ? & ? & ? \end{pmatrix}. \tag{2.20}$$

According to the inverse calculations shown above, there seems to be one answer in Equation 2.20: *strawberry*.

Generating the Word

The matrix is 4×4 large, which means that the word can be 16 characters long. The generator contains many words that can be randomly chosen.

```
1 private static final List<String> words =
     Arrays.asList("strawberry" /*, ...*/);
2 private final Random secureRandom = new SecureRandom();
3
4 public WordData generate() {
5   int idx = secureRandom.nextInt(words.size());
6   String word = words.get(idx);
7
8   return new WordData(word);
9 }
```

Partial Encryption of the Word

Generating and encrypting data are separate functionalities. While the generating is not such attractive, the implementation of partial encryption is worth noting. However, the AES methods have many in common. Their input is a 16byte array, and their output type is the same as the input. In this situation, it is possible to consider a design pattern. One basic and frequently used is Factory, which is described in Figure 2.14 and Listing 2.1.

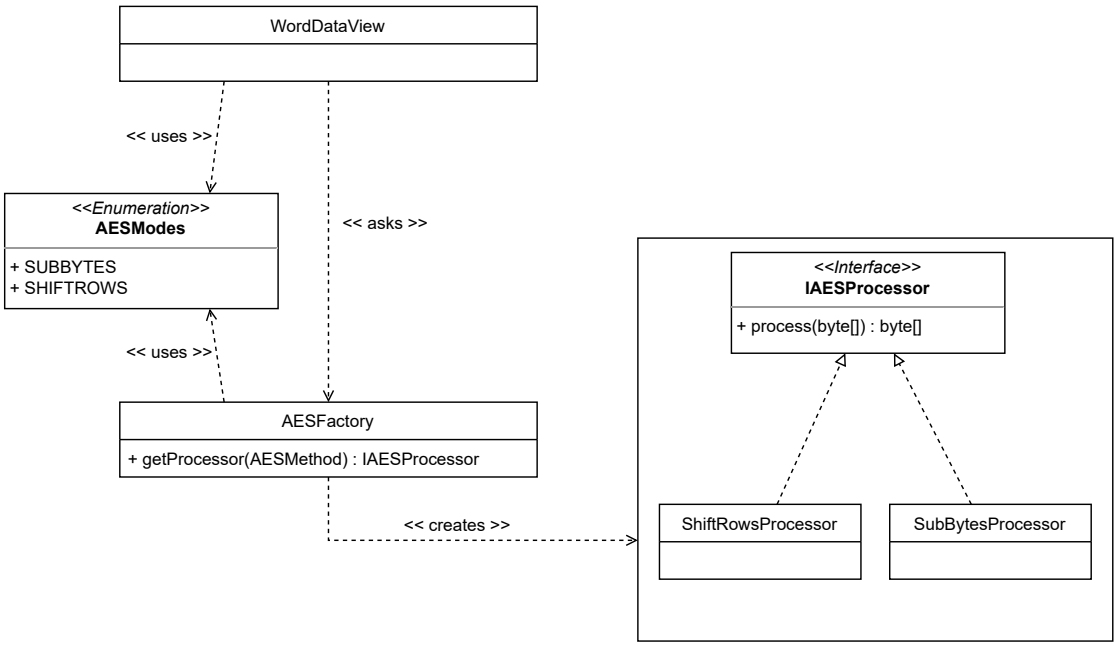


Fig. 2.14: Diagram of factory pattern

Listing 2.1: Factory implementation in Java

```
@NoArgsConstructor(access = AccessLevel.PRIVATE)
public class AESFactory {

  public static IAESProcessor getProcessor(AESModes modes) {
    switch (modes) {
      case SUBBYTES:
        return new SubBytesProcessor();

        case SHIFTROWS:
        return new ShiftRowsProcessor();

      default:
        throw new IllegalArgumentException("Mode " + modes + " is
            not supported.");
    }
  }
}

public enum AESModes {
  SUBBYTES,
  SHIFTROWS;
}

public interface IAESProcessor {
  byte[] process(byte[] data);
}

public class SubBytesProcessor implements IAESProcessor {

  private static final int MAX_LENGTH = 16;

  @Override
  public byte[] process(byte[] data) {
    byte[] result = new byte[MAX_LENGTH];

    //process

    return result;
  }
}
```

2.4.3 Room Charlie

Room Charlie is the next room after Bravo, which focuses on the complete encryption and decryption of user data. Some kind of steganography and social engineering also reflects here. The room is set, seemingly, in a gallery. In Figure 2.15, there is an extraordinary picture, which may not seem like a picture, a certificate, and a security guard.

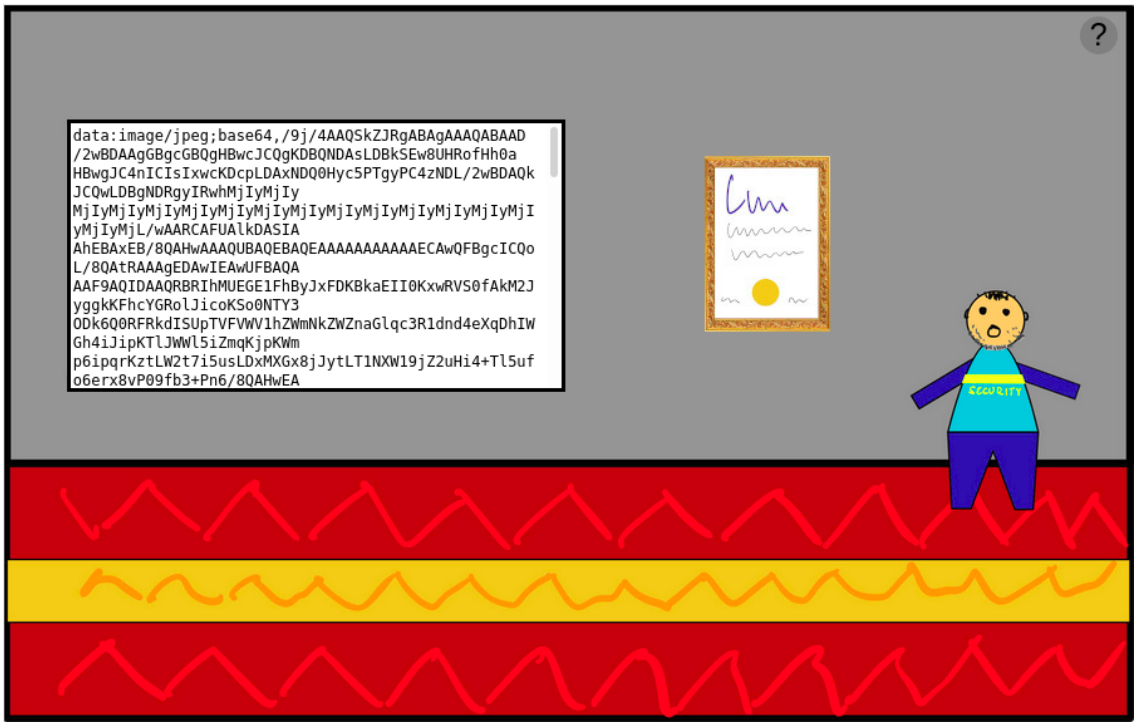


Fig. 2.15: Room charlie

The Picture

There is no discussion that the painting on the wall is not in its usual form. The attentive have certainly noticed that this is base64 encoding. It is possible to encode any byte sequence into base64, unlike ordinary text, which could be noticed in the previous chapter in Equation 2.19, where non-alphabetic characters are hard to display in text form.

Furthermore, images are not anything special. As well as any text, an image is an ordered sequence of bytes. So any picture can be encoded into base64. Many web browsers (tested Mozilla Firefox and Google Chrome) can decode base64 data into an image. Figure 2.16 is the decoded image using Mozilla Firefox. From the picture, it is possible to read a secret: `LjiLxxe7ll0C2tht`, which will be needed in the next step.

Fig. 2.16: Decoded image

Appending the text to the image is performed in an easy class using the `java.awt` package, as shown in Listing 2.2.

Listing 2.2: Text appender

```
1  public class TextAppender {
2
3    @SneakyThrows
4    public File add(String string) {
5      // the path is edited due to it is too long
6      final BufferedImage image = ImageIO.read(new
           File("image.jpg"));
7
8      Graphics g = image.getGraphics();
9      g.setFont(g.getFont().deriveFont(10f));
10     g.drawString(string, 10, 20);
11     g.dispose();
12
13     File file = File.createTempFile("text-appender", ".jpg");
14     ImageIO.write(image, "jpg", file);
15
16     return file;
17   }
18 }
```

Then the image has to be encoded into base64; this is done using the `java.nio.Files` and `java.util.Base64`, as shown in Listing 2.3.

Listing 2.3: Image Base64 encoder

```
public class Image {

  @SneakyThrows
  public String load(File file) {
    byte[] bytes = Files.readAllBytes(file.toPath());

    String data = Base64.getMimeEncoder()
          .withoutPadding()
          .encodeToString(bytes);

    return "" + data;
  }
}
```

And finally the output of this method is edited and encoded image. Example of usage is shown in Listing 2.4.

Listing 2.4: Example of usage

```
String generatedCertKey = new WordPasswordGenerator();
File passwordFile = new TextAppender().add(generatedCertKey);
String imgBase64 = new Image().load(passwordFile);
```

Certificate

An inconspicuous-looking certificate on the hallway wall may seem unnecessary. However, after clicking on it, a file named `secret.enc.base64` will start downloading. The file contains, as the name implies, an encoded text, which is listed in Listing 2.5. When decoding, its output is nonsense, it means that the text is encrypted. And it's time for the text from the picture on the wall.

Listing 2.5: Encrypted text

```
q5hA0X6wCLehRLhHIz2gXeb5i1Ow+wh6PvQDusCfY8s=
```

Now we have two elements, one resembling a secret key and the other encrypted text. Let us try to decrypt it. There are many web applications[4] on the Internet for encrypting and decrypting text.

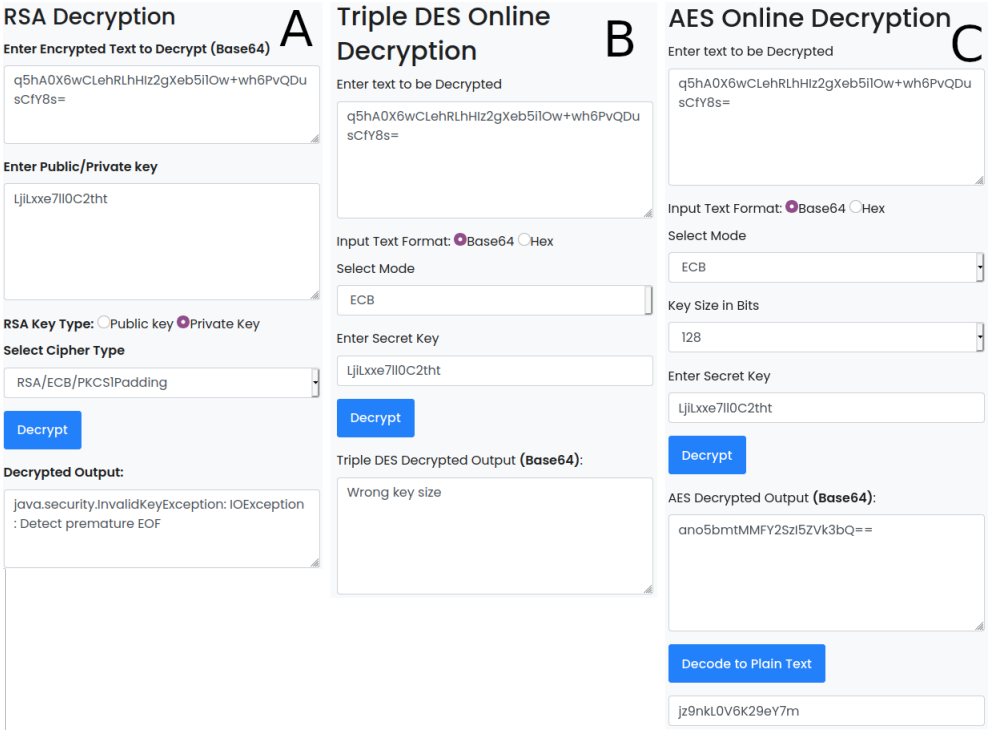


Fig. 2.17: Decryption

In the first case (A) of Figure 2.17, we tried RSA, which failed because of invalid key. In the case (B), it also failed, 3DES uses 56bits long secret key, ours is 128bits long (128 b = 16 B ∼ 16 characters). Finally, the AES128 decryption ECB, figure (C), gave us some result, that is `jz9nkL0V6K29eY7m`. Let us continue.

Security Guard

After an interaction and telling the security guy a secret password, he let the user go to the last room. However, as often happens, people may be the weakest link in the system. With a bit of social engineering, even in this case, it is possible to get to the next room without solving tasks.

After several attempts to annoy the security guard, the chance of being let into another room increases. The probability is 10 percent for each interaction with the security guard after the first 20 attempts. The interaction is shown in Figure 2.18.

[4]Different Online Crypto Tools – `https://www.devglan.com/cryptotools/cryptography-tools`

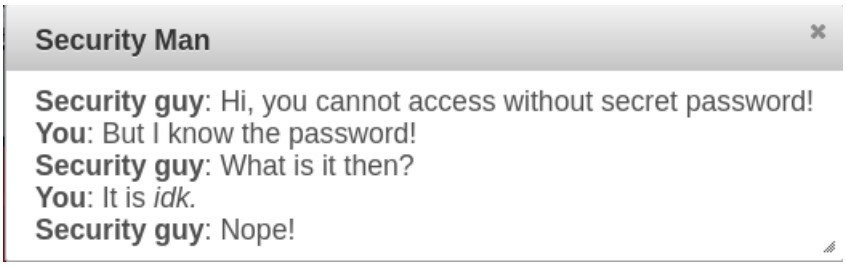


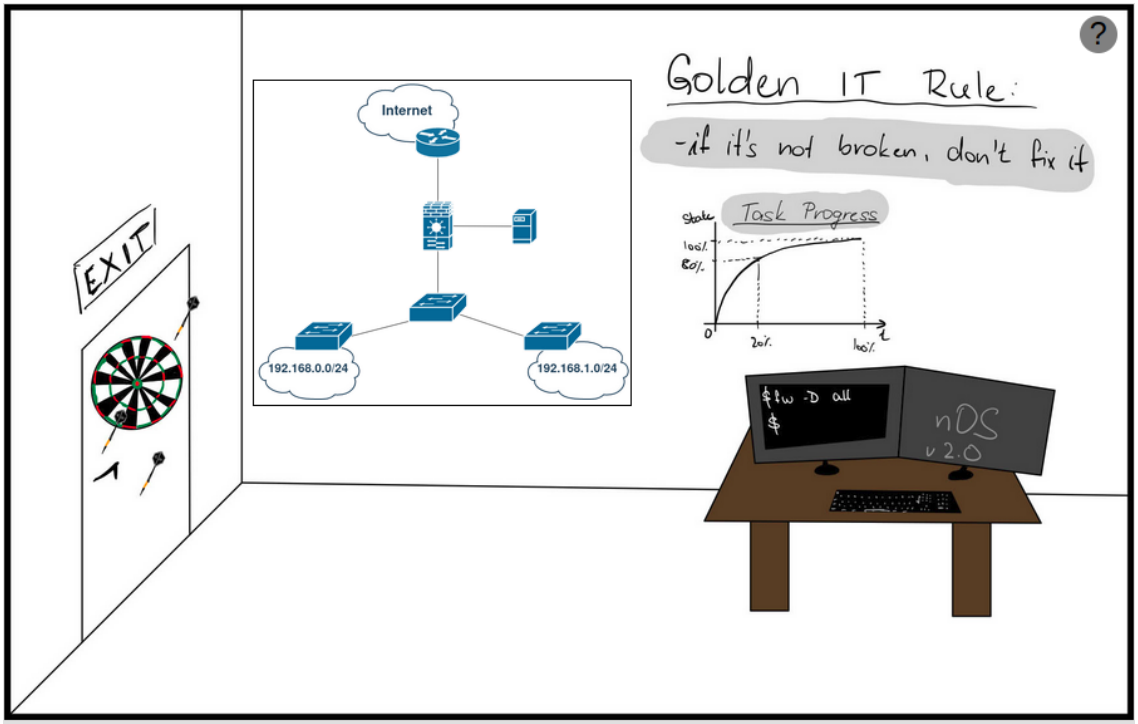
Fig. 2.18: Example of interaction

In fact, we have already found out the password. After telling the security guard, the application is redirected to the last room.

2.4.4 Room Delta

Room Delta is the fourth and final room in the game. The user's task is to block an ongoing attack. The only way to do so is a computer with a terminal and a few applications. One of them is the firewall.

In Figure 2.19, the fourth room environment is designed as a network administrator's poor office with a few elements. One of the elements is a computer network diagram, a dartboard exchanged for a chair with no single dart in the target, and the mentioned computer with a terminal.



There is an ongoing attack on your company! Use your computer with terminal to block the attackers! After your job is done, run command *testfw*.

Fig. 2.19: Room delta

Terminal

After clicking on the computer, a dialog window appears, which represents the terminal. The terminal window demonstrates Figure 2.20.

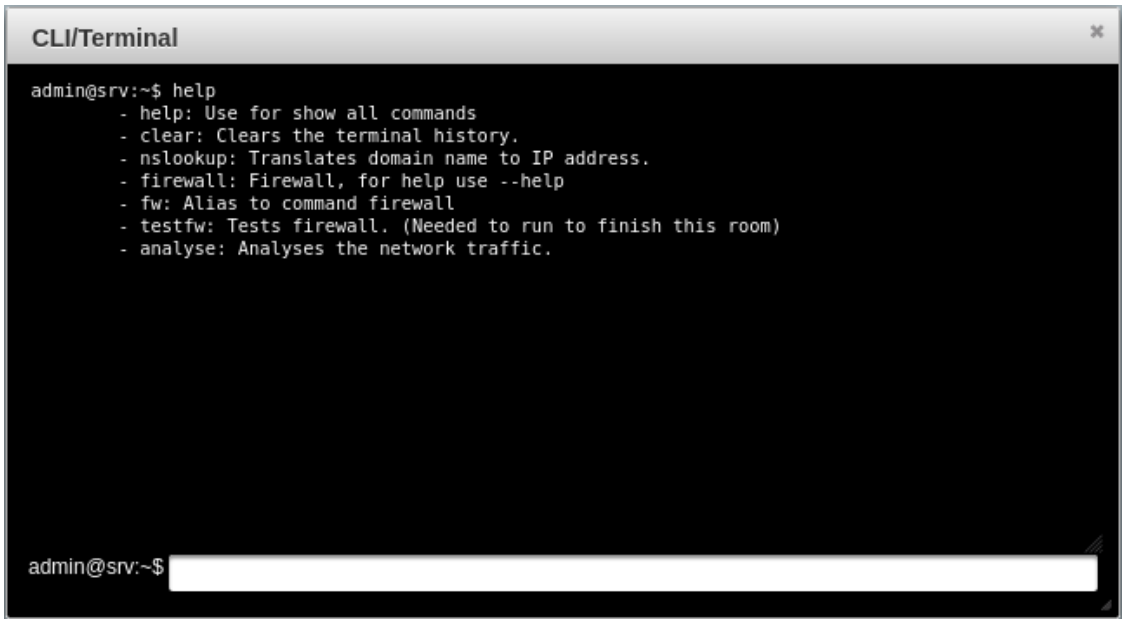


Fig. 2.20: Terminal

The terminal tries to simulate the behavior as a standard terminal on Linux systems. Many features are common. Users can use up and down keys to navigate in the command history. Along with the history, the whisperer with TAB key works too. When the characters sequence is unique enough to detect the command, pressing the TAB completes the command. For example, `ana` completes to `analyse`.

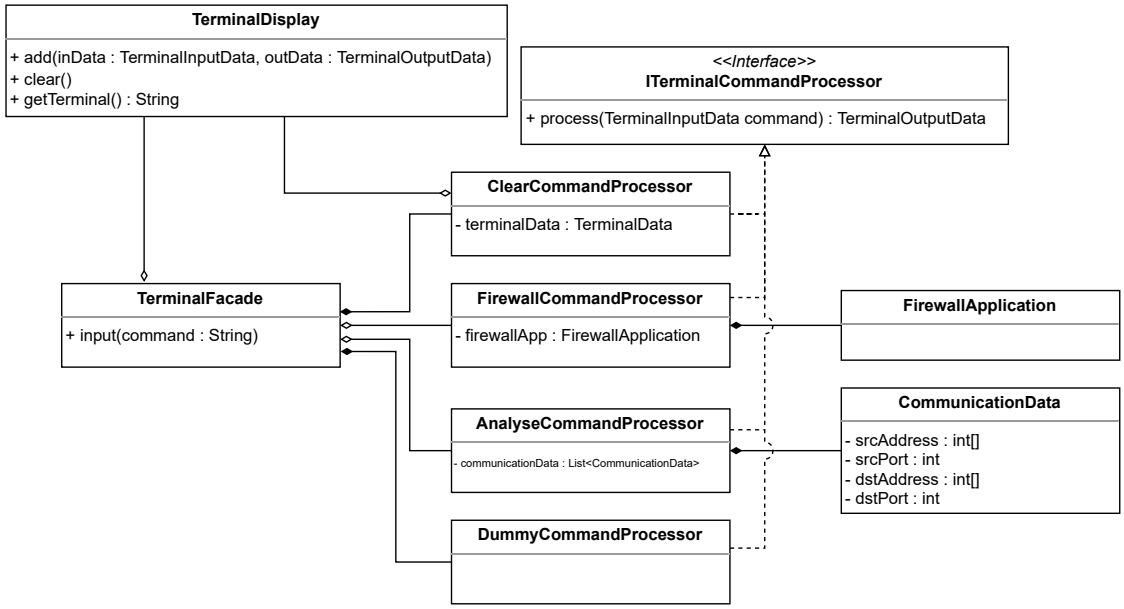


Fig. 2.21: Diagram of terminal

Figure 2.21 describes that **TerminalFacade** provides communication with the terminal service, chooses the responsible class, and lets it to processes the command. After the command is processed, the output is sent to TerminalDisplay.

TerminalDisplay is the class that stores inputs and outputs of the terminal window. The class is injected into the TerminalFacade.

ITerminalCommandProcessor is an interface, which defines the API of the command processor.

ClearCommandProcessor and all other similar classes are concrete implementations for handling the command. Each command has its class and one dummy that handles unknown command, this also corresponds to a Null Pattern. Listing 2.6 demonstrates command handling.

Listing 2.6: Dummy processor

```
1  public class DummyCommandProcessor implements ITCP {
2    @Override
3    public TOD process(TID inputData) {
4      String command = inputData.getCommand();
5      String response = "Command " + command + " not found.";
6      TOD outputData = new TOD();
7      outputData.setResponse(response);
8      return outputData;
9    }
10 }
```

Firewall Application

Generally, a firewall is an application that filters communication based on created rules. In this case, the firewall can check communication on the third and fourth layers of the OSI model, respectively source address and port and destination address and port.

The firewall application implementation can be split into two parts: a set of records and testers. Records store rules with information, which traffic to accept or drop. Meanwhile, the testers test traffic and block the traffic.

One record may contain this information: a source address, a source wildcard mask, source ports, a destination address, a destination wildcard mask, and destination ports.

In the binary form of wildcard mask, 0 in a position means that the rule address and tested address must match at the given position. Example 2.1 solves a simulated situation of address comparison.

rule address: 192.168.1.8,
wildcard mask: 0.0.0.1,
tested address: 192.168.1.9

The wildcard mask is zero in the first 3 bytes. That means the tested address must match the first bytes. And it does. However, how does it work in the fourth byte? Do some math. Firstly, the rule address and tested address are XORed, and then the result ANDs with the negated wildcard mask:

$$9_{(10)} \oplus 8_{(10)} = 1_{(10)}, \text{ which is}$$
$$1001_{(2)} \oplus 1000_{(2)} = 0001_{(2)}$$

and then

$$1_{(10)} \wedge \neg 1_{(10)} = 0_{(10)}, \text{ which equals to}$$
$$0001_{(2)} \wedge \neg 0001_{(2)} = 0000_{(2)}, \text{ which equals to}$$
$$0001_{(2)} \wedge 1110_{(2)} = 0000_{(2)}.$$

When the result is zero, then the tested address matches the prescripted address.

Example 2.1: Wildcard calculation

Solving the Task

The main commands are: `analyse`, `firewall` or `fw` and `testfw`. While the analyze command prints a network analysis, the firewall command sets up traffic filtering based on several parameters. As described above, this is filtering based on the source or destination IP address and port.

Listing 2.7: Output of network analysis

```
admin@srv:~$ analyse
Increased traffic has been detected:
  214.5.176.135:33089 -> 192.168.2.49:80
  105.195.156.215:53382 -> 192.168.2.49:80
  196.156.51.36:45155 -> 192.168.2.49:80
  114.218.243.145:37276 -> 192.168.2.49:443
  163.127.140.55:31749 -> 192.168.2.49:53
```

The output in Listing 2.7 shows that the attack directs the server with IP address 192.168.2.49 on its essential services: WEB services and DNS. Probably it is a DDoS attack attempt. It is unknown how many requests head to the server from the listing, but the addresses have been evaluated as risky. And the firewall should block them.

Firstly, let us find out information how to use the firewall command. Simply typing the command prompts us to enter the parameter `--help`, which prints methods and parameters to use. Whereas parameters in the brackets are optional, the parameters in angle brackets are required. Additionally, there is important information about the default rule at the end of Listing 2.8 when none of the firewall records applies.

Listing 2.8: Firewall help

```
admin@srv:~$ firewall
Try firewall --help or -H.

admin@srv:~$ firewall --help
  -A|--append [--src-address <ip_address> [wildcard_mask]]
     [--dst-address <ip_address> [wildcard_mask]] --action
     <ACCEPT/DROP>
  -S|--show Shows all records
  -D|--delete <row>

When none of the rules matches, the ACCEPT ALL rule is applied.
```

Before creating new rules, it is necessary to find out the current state of the firewall. While designing new network policies, the administrator should know the basics about the system. Without the knowledge, the administrator can cause many problems. Now we know that we need to use switch `--show` or `-S` to print the rules.

Listing 2.9: Firewall records

```
admin@srv:~$ fw -S

#  ACTION SRC ADDR        SRC WLD MASK  DST ADDR        DST WLD
   MASK
```

Listing 2.9 shows that there are not any records, and all network traffic is allowed without any restrictions. Moreover, it is possible to use the alias `fw` for running the `firewall` command.

The firewall is clean, and the attack continues. It is time to consider a strategy with the technologies we have and design new network rules. To protect against the unsolicited data flow, we can completely cut off the server from the network not to lose data. But this causes that the services won't be able to our intranet users and users from the Internet.

Another option is to deny access to the Internet and from it. Intranet services will work, and rules will block the attack. The only not minor problem is that the users won't be able to use the Internet, which is unwanted.

Last but not least, the direct blocking of attackers. This will deny the attack, and all services for intranet clients and customers from outside the network will be operational. Only if another attack occurs will you need to create additional rules. However, we cannot solve this otherwise with current technologies.

Listing 2.10 shows the insertion of a few of records based on the source IP address.

Listing 2.10: Inserting all the attacking IP addresses

```
admin@srv:~$ fw -A -s 214.5.176.135 -d 192.168.2.49 -a DROP
Rule successfully added.

<< the output is reduced >>

admin@srv:~ fw -A -s 163.127.140.55 -d 192.168.2.49 -a DROP
Rule successfully added.

admin@srv:~ fw -S

#  ACTION SRC ADDR        SRC WLD MASK  DST ADDR        DST WLD
   MASK
0  DENY   214.5.176.135:-- 0.0.0.0     192.168.2.49:-- 0.0.0.0
1  DENY   105.195.156.215:-- 0.0.0.0   192.168.2.49:-- 0.0.0.0
2  DENY   196.156.51.36:-- 0.0.0.0     192.168.2.49:-- 0.0.0.0
3  DENY   114.218.243.145:-- 0.0.0.0   192.168.2.49:-- 0.0.0.0
4  DENY   163.127.140.55:-- 0.0.0.0    192.168.2.49:-- 0.0.0.0
```

When the firewall is set up, we can run the `testfw` command. This checks the primary connections and moves us next. More about tests explains chapter Firewall Tests. The results of the tests are shown in Listing 2.11.

Listing 2.11: Output of tests

```
admin@srv:~$ testfw
Test result:
  Test no 1: ✓
  Test no 2: ✓
  Test no 3: ✓
```

After running the `testfw` command and all the tests pass, the application redirects to final page.

Firewall Tests

The firewall tests check correct firewall settings. There are three sets of tests. Each handles a different area. The first set verifies that all attack messages are filtered out. The second and third sets ascertain the server's availability, respectively, the Internet connectivity, from the internal network. In both cases, the tests check connectivity requests and the response. Graphically displayed test sets are shown in Figure 2.22.

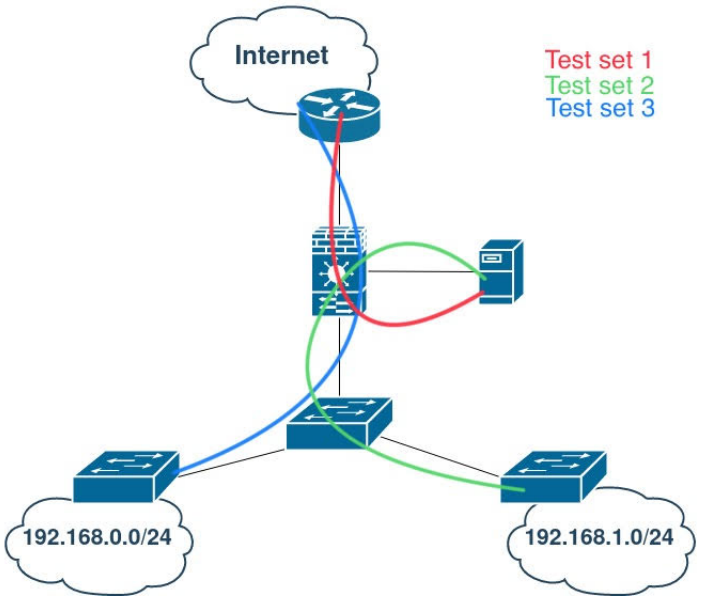


Fig. 2.22: Network diagram

Local and global connection tests randomly generate IP addresses. For local clients, it is from the supernet of `192.168.0.0/23`. For the global clients, the first byte is from the range 50–230 with filtered supernets: `127.0.0.0/8`, `10.0.0.0/8`, `192.0.0.0/8`, `172.0.0.0/8` and `224.0.0.0/8`. Each test set contains 10 pairs of random connections, Listing 2.12 shows two of them.

Listing 2.12: Example of generated tests

```
// local tests
192.168.1.113:23042 -> 192.168.2.49:443
192.168.2.49:443 -> 192.168.1.113:23042

// global tests
192.168.1.228:36496 -> 154.133.207.32:443
154.133.207.32:443 -> 192.168.1.228:36496
```

2.5 HTTPS

The application has been secured using a TLS certificate. The certificate was generated with Certbot, [25] a free and open-source tool for generating Let's Encrypt certificates. Let's Encrypt [26] is a free service that provides certificates signed by Certificate Authority (CA). Internet Security Research Group provides the certificate service. The main benefits [26] are free, automatic, and quick obtaining the certificate.

Creating the certificate

Firstly, we need to download the Certbot software. This was done using snap package manager. Listing 2.13 shows command to download Certbot with snap.

Listing 2.13: Download certbot

```
$ sudo snap install --classic certbot
```

The next step is to run the Certbot and generate a certificate. We have to make that no service is using HTTP port 80. And then we proceed to the generation. The software asks us to provide domains for the certificate. The steps continue in Listing 2.14.

Listing 2.14: Generate a certificate

```
$ sudo certbot certonly --standalone

Please enter in your domain name(s) (comma and/or space
   separated) (Enter 'c' to cancel): nosek.fun
Requesting a certificate for nosek.fun
Performing the following challenges:
http-01 challenge for nosek.fun
Waiting for verification...
Cleaning up challenges

IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved
    at:
   /etc/letsencrypt/live/nosek.fun/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/nosek.fun/privkey.pem
   Your certificate will expire on 2021-08-27.
```

For the Spring integration, we have to create PKCS #12 file which contain many cryptographic files. We are going to use openssl software. The steps continue in Listing 2.15.

Listing 2.15: Creating PKCS12 file

```
$ openssl pkcs12 -export -in fullchain.pem \
-out keystore.p12 \
-inkey privkey.pem \
-name spring \
-CAfile chain.pem \
-caname root
Enter Export Password: <password>
Verifying - Enter Export Password: <password>
```

Now we have the PKCS #12 file. Now we have to configure the web server to use the certificate. The configuration shows Listing 2.16.

Listing 2.16: Spring HTTPS configuration

```
server.port=443
security.require-ssl=false
server.ssl.key-store=/etc/letsencrypt/live/nosek.fun/keystore.p12
server.ssl.key-store-password=<password>
server.ssl.keyStoreType=PKCS12
server.ssl.keyAlias=spring
```

And the HTTPS is running now. Figure 2.23 shows the certificate.

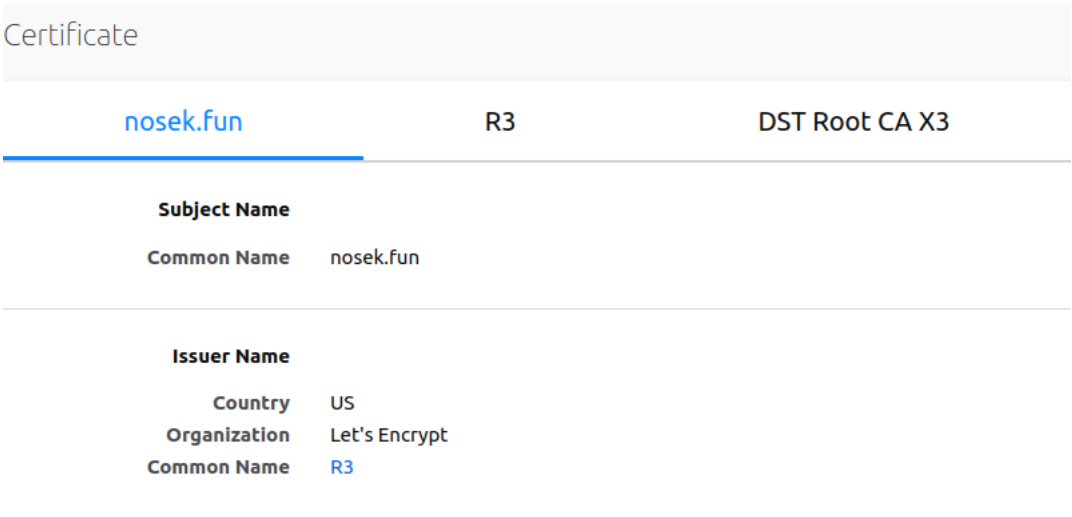


Fig. 2.23: Secured connection

Conclusion

In the theoretical introduction, the basic terms such as cryptography, web application, escape game were explained, which were the central theme for the application. Furthermore, the chapters explained the technologies used for application development. The text shows a web application built on the Java platform using the Spring Framework with JakartaServer Faces and PrimeFaces technology for the user interface. Gradle and Git were also used in the implementation.

The beginning of the implementation chapter mentions a four-tier software architecture with a description of each of them and further describes application progress across four rooms. Pictures of Spring Intializr tool show how to create a Spring Framework including other Spring modules easily. The chapter also mentions the simple creation of a logo without any graphics software. The rest deals with the solution of all four rooms with reference to interesting algorithmic problems.

Thus, a web escape game was implemented to solve cryptographic challenges: modular arithmetic, AES algorithm, decryption, and network security in the form of a firewall. The application architecture facilitates changes or assignments to new rooms and application properties, such as a point system. During the work on this work, there were regular consultations with the leader. However, this fulfilled the requirements in the assignment of the bachelor's thesis.

The application runs on a VPS, which is accessible via `nosek.fun`, and everyone can try to play it. The server uses HTTPS, so the connection to the application is secured.

Bibliography

[1] SCHÖNWÄLDER, Jürgen. 2002. *Evolution of Open Source SNMP Tools*.

[2] GAMAGE, Thilina Ashen. 2017. *Evolution of HTTP — HTTP/0.9, HTTP/1.0, HTTP/1.1, Keep-Alive, Upgrade, and HTTPS*. Medium [online]. Available at: https://medium.com/platform-engineer/evolution-of-http-69cfe6531ba0

[3] *HTTPS encryption on the web. Google Transparency Report* [online]. 14. listopad 2020 [cit. 2020-12-11]. Available at: <https://transparencyreport.google.com/https/overview>.

[4] OSTEBEE, A., 1997. *Handbook of Applied Cryptography*. The American Mathematical Monthly, Aug, vol. 104, no. 7, pp. 683 ProQuest Central. ISSN 00029890.

[5] *What is Asymmetric Encryption? Understand with Simple Examples. Cheap SSL Security* [online]. [cit. 2020-12-11]. Available at: <https://cheapsslsecurity.com/blog/what-is-asymmetric-encryption-understand-with-simple-examples/>.

[6] BORISOV, Nikita, Ian GOLDBERG a David WAGNER. *Intercepting Mobile Communications.* University of California, Berkeley: The Insecurity of 802.11 [online]. Berkeley, California [cit. 2020-12-11]. Available at: <http://www.isaac.cs.berkeley.edu/isaac/wep-draft.pdf>.

[7] THOMPSON, Eric. *MD5 collisions and the impact on computer forensics* [online]. 2005, (1), 36 - 40 [cit. 2020-12-11]. ISSN 1742-2876. Available at: <doi:https://doi.org/10.1016/j.diin.2005.01.004>.

[8] MKS075. *Difference between Block Cipher and Stream Cipher*. GeeksforGeeks [online]. 07-07-2020 [cit. 2020-12-11]. Available at: <https://www.geeksforgeeks.org/difference-between-block-cipher-and-stream-cipher/>.

[9] ROSSI, Gustavo, Oscar PASTOR, Daniel SCHWABE and Luis OLSINA. 2007. *Web engineering: modelling and implementing web applications*. Springer Science & Business Media.

[10] VERGNE, Matthew J., Joshua D. SIMMONS a Ryan S. BOWEN, 2019. *Escape the Lab: An Interactive Escape-Room Game as a Laboratory Experiment.* Journal of Chemical Education. **96**(5), 985-991. ISSN 0021-9584. Available at: doi:10.1021/acs.jchemed.8b01023

[11] *The Java Language Environment. Oracle* [online]. [cit. 2020-12-11]. Available at: <https://www.oracle.com/java/technologies/performance-comparisons.html>.

[12] CHAN, Rosalie. *The 10 most popular programming languages, according to the ‚Facebook for programmers'.* Business Insider [online]. 2019, 22-01-2019 [cit. 2020-12-11]. Available at: <https://www.businessinsider.de/international/the-10-most-popular-programming-languages-according-to-github-2018-10/?op=1>.

[13] Oracle Announces Java 16, 2021. *Oracle* [online]. Austin, Texas [cit. 2021-5-29]. Available at: https://www.oracle.com/news/announcement/oracle-announces-java-16-031621.html

[14] *Spring Framework Overview* [online], 2021. [cit. 2021-5-30]. Available at: https://docs.spring.io/spring-framework/docs/current/reference/html/overview.html

[15] MAJUMDAR, Rana, Rachna JAIN, Shivam BARTHWAL, Chetna BARTHWAL and CHOUDHARY. 2017. *Source code management using version control system.* 2017. Noida, India: IEEE. ISBN 978-1-5090-3012-5.DOI 10.1109/ICRITO.2017.8342438

[16] *Compare Repositories. Open Hub* [online]. [cit. 2020-12-11]. Available at: <https://www.openhub.net/repositories/compare>.

[17] *Developer Survey Results 2018: Version Control. Stack Overflow* [online]. 2018 [cit. 2020-12-11]. Available at: <https://insights.stackoverflow.com/survey/2018#work-_-version-control>.

[18] KARANAM, Ranga. *Best Java Unit Testing Frameworks.* DZone: Java Zone [online]. 2019, 12 September 2019 [cit. 2020-12-11]. Available at: <https://dzone.com/articles/best-java-unit-testing-frameworks>.

[19] *Build automation. In: Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-12-11]. Available at: <https://en.wikipedia.org/wiki/Build_automation>.

[20] *Apache Maven. In: Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-12-11]. Available at: <https://en.wikipedia.org/wiki/Apache_Maven>.

[21] MAPLE, Simon a Andrew BINSTOCK. *JVM Ecosystem report 2018: About your Tools*. Snyk [online]. 2018, 17 October 2018 [cit. 2020-12-11]. Available at: <https://snyk.io/blog/jvm-ecosystem-report-2018-tools/>.

[22] *Gradle vs Maven Comparison. Gradle* [online]. [cit. 2020-12-11]. Available at: <https://gradle.org/maven-vs-gradle/>. https://www.javatpoint.com/gradle-vs-maven

[23] *Gradle vs. Maven. Javatpoint* [online]. [cit. 2020-12-11]. Available at: <https://www.javatpoint.com/gradle-vs-maven>.

[24] *JetBrains: Essential tools for software developers and teams* [online]. [cit. 2021-05-27]. Available at: <hthttps://www.jetbrains.com/>.

[25] *Certbot - About Certbot* [online]. [cit. 2021-5-30] Available at: https://certbot.eff.org/about

[26] *About Let's Encrypt - Let's Encrypt* [online]. [cit. 2021-5-30] Available at: https://letsencrypt.org/about/

List of symbols, quantities and abbreviations

3DES	Triple Data Encryption Standard
ADDR	Address
AES	Advanced Encryption Standard
API	Application Programming Interface
CDI	Context Dependency Inject
DDoS	Distributed Denial of Service
DES	Data Encryption Standard
DH	Diffie-Hellman
DNS	Domain Name System
DST	Destination
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
IP	Internet Protocol
ITCP	Interface Terminal Command Processor
J2EE	Java 2 Enterprise Edition
JEE	Java/Jakarta Enterprise Edition
JPA	Java Persistence API
JPQL	Java Persistence Query Language
JRE	Java Runtime Environment
JS	JavaScript

JSF	JavaServer/JakartaServer Faces
JVM	Java Virtual Machine
MD5	Message-Digest Algorithm version 5
MVC	Model-View-Controller
PHP	Hypertext Preprocessor
PIN	Personal Identification Number
RC4	Rivest Cipher 4
REST	Representational State Transfer
RSA	Rivest–Shamir–Adleman
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
SRC	Source
SVN	Subversion
TID	Terminal Input Data
TOD	Terminal Output Data
UI	User Interface
VCS	Version Control System
WEP	Wired Equivalent Privacy
WLD	Wildcard
WWW	World Wide Web

A Application Launch

The application is publicly accessible on `nosek.fun`. Although, you can run your own instance. Firstly, it is necessary to have Java 11 JRE or later on your computer. You can verify it using command `java`. It is important to have an access to the internet. An example of output is in Listing A.1.

Listing A.1: Java version

```
$ java -version
openjdk version "11.0.11" 2021-04-20
```

After verified Java version, we move to the directory with exported source codes and build the project as Listing A.2 shows.

Listing A.2: Run the application

```
$ ./gradlew build
<output omitted>
BUILD SUCCESSFUL in 13s
8 actionable tasks: 8 executed
```

The move to directory `build/libs`. And run the application as describes Listing A.3.

Listing A.3: Run the application

```
$ cd build/libs/
$ java -jar erg-1.0.war
```

We can add some properties, for example, server port, SSL settings or logging level, which describes Listing A.4.

Listing A.4: Run the application

```
$ java -jar erg-1.0.war \
--server.port=443 \
--cz.vutbr.xnosek12.erg.state=PRODUCTION \
--security.require-ssl=false \
--server.ssl.key-store=<PKCS12-file-path> \
--server.ssl.key-store-password=<cert-password> \
--server.ssl.keyStoreType=PKCS12 \
--server.ssl.keyAlias=spring \
--logging.level.cz.vutbr.xnosek12=INFO
```

In case you want to use your own database, see Database chapter.

A.1 Database

```
1 CREATE DATABASE erg;
2
3 CREATE USER 'erg'@'%' INDETIFIED BY 'XF7z5mVDaZvvh6tt';
4
5 GRANT SELECT, INSERT ON erg.* to 'erg'@'%';
6
7 CREATE TABLE erg.stats (id int(6) UNSIGNED AUTO_INCREMENT
    PRIMARY KEY, nickname VARCHAR(30) NOT NULL, time BIGINT);
```

B Attachment “src”

The attachment `src.zip` contains Gradle project with source codes importable to any IDE.

- `erg` ..... Project root
  - `gradle` ..... Gradle application directory
    - `(...)`
  - `src` ..... Source codes
    - `(...)`
  - `settings.gradle`
  - `gradlew`
  - `gradlew.bat`
  - `build.gradle` ..... Gradle project configuration file
  - `HELP.md`